1 слайд

Параллельная реализация метода эллипсоидов для задач оптимизации большой размерности

Безбородов В.А.

ФГБОУ ВПО ЮУрГУ
г. Челябинск

10 мая 2015 г.

2 слайд

Содержание

3 слайд

Метод эллипсоидов предложили

1976 **Юдин Д.Б. и Немировский А.С.** как метод последовательных отсечений.

1977 **Шор Н.З.** как вариант метода с растяжением пространства в направлении субградиента.

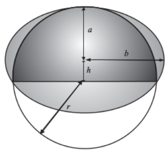
1979 **Хачиян Л.** построил первый полиномиальный алгоритм решения задачи ЛП с рациональными коэффициентами.

4 слайд

Содержание

5 слайд

1-d эллипсоид и его свойства



Эллипсоид ε_n, содержащий полушар в E_n, имеет параметры
$$b = \left(\alpha + \frac{1}{\alpha}\right)\frac{r}{2}, \quad h = \left(1 - \frac{1}{\alpha^2}\right)\frac{r}{2},$$
где $\alpha = \frac{b}{a}$ и r – радиус шара S_n.

Если пространство «растянуть» с коэффициентом α в направлении полуоси a, то ε_n станет шаром в преобразованном пространстве.

Отношение объема эллипсоида ε_n к объему шара S_n равно
$$q(n) = \frac{vol(\varepsilon_n)}{vol(S_n)} = \frac{1}{\alpha}\left(\frac{b}{r}\right)^n = \frac{1}{\alpha}\left(\frac{1}{2}\left(\alpha + \frac{1}{\alpha}\right)\right)^n.$$

6 слайд

Содержание

7 слайд

Метод эллипсоидов | Алгоритм метода

Использование метода эллипсоидов

МЭ используется для решения следующей задачи

На E_n ($n \geq 1$) определено векторное поле $g(x)$, $g(x) \in E_n$. Требуется найти точку x^*, такую, что $(g(x), x - x^*) \geq 0$ для всех $x \in E_n$. Предполагается, что x^* существует и $g(x) \neq 0$ для $x \neq x^*$.

К такой задаче сводятся задачи математического программирования:

- задача безусловной минимизации выпуклой функции,
- общая задача выпуклого программирования,
- задача о седловой точке выпукло-вогнутых функций.

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 7 / 21

8 слайд

Метод эллипсоидов | Алгоритм метода

Стартовые условия для МЭ

Задан:

Коэффициент α такой, что $\alpha + 1/\alpha < 2\sqrt[n]{\alpha}$.

Инициализация:

1. Выбрать стартовую точку $x_0 \in E_n$ и начальный радиус r_0, такой что $||x_0 - x^*|| \leq r_0$.
2. Положить $B_0 := E$, где E – единичная матрица.

Перейти к очередной итерации со значениями x_0, r_0, B_0.

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 8 / 21

9 слайд

Метод эллипсоидов | Алгоритм метода

Алгоритм

Пусть на k-й итерации найдены $x_k \in E_n$, r_k и B_k. Для перехода к $(k+1)$-й итерации выполнить:

Шаг 1. Вычислить $g(x_k)$. Если $g(x_k) = 0$, то **ОСТАНОВ**$(x^* = x_k)$.

Шаг 2. Вычислить очередную точку $x_{k+1} = x_k - h_k B_k \xi_k$, где $h_k = \frac{1}{2}\left(1 - \frac{1}{\alpha^2}\right)$, $\xi_k = \frac{B_k^T g(x_k)}{||B_k^T g(x_k)||}$.

Шаг 3. Пересчитать матрицу B_{k+1} и радиус r_{k+1}

$B_{k+1} = B_k + (\beta - 1)(B_k \xi_k)\xi_k^T$, $\beta = \frac{1}{\alpha}$,

$r_{k+1} = (\alpha + \beta)\frac{r_k}{2}$.

Шаг 4. Перейти к $(k+1)$-й итерации с x_{k+1}, r_{k+1} и B_{k+1}.

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 9 / 21

10 слайд

Метод эллипсоидов | Алгоритм метода

О сходимости метода эллипсоидов

Теорема (Локализация x^* в эллипсоиде)

Генерируемая методом эллипсоидов последовательность точек $\{x_k\}_{k=0}^{\infty}$ удовлетворяет неравенствам

$$||B_k^{-1}(x_k - x^*)|| \leq r_k, \quad k = 0, 1, 2, \ldots$$

Следствие

Эллипсоид $\varepsilon_k = \{x : ||B_k^{-1}(x_k - x)|| \leq r_k\}$ *содержит точку* x^*.

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 10 / 21

11 слайд

Метод эллипсоидов | Алгоритм метода

О сходимости метода эллипсоидов

Теорема (О скорости сходимости)

Для всех итераций метода эллипсоидов коэффициент уменьшения объема эллипсоида, локализующего x^*, есть величина постоянная и равная

$$q(n) = \frac{vol(\varepsilon_{k+1})}{vol(\varepsilon_k)} = \frac{1}{\alpha}\left(\frac{1}{2}\left(\alpha + \frac{1}{\alpha}\right)\right)^n < 1, \quad k = 0, 1, 2, \ldots$$

Оптимальный коэффициент растяжения пространства

$$\alpha = \sqrt{\frac{n+1}{n-1}} \Rightarrow q(n) = \sqrt{\frac{n-1}{n+1}}\left(\frac{n^2}{n^2-1}\right)^{n/2} \leq 1 - \frac{1}{2n}.$$

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 11 / 21

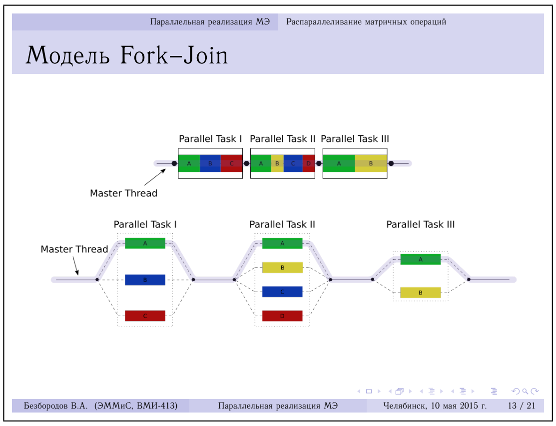
12 слайд

Параллельная реализация МЭ | Распараллеливание матричных операций

Содержание

Безбородов В.А. (ЭММиС, ВМИ-413) Параллельная реализация МЭ Челябинск, 10 мая 2015 г. 12 / 21

13 слайд



В параллельном программировании, Fork-Join model (модель ветвление-объединение, FJM) – это способ запуска и выполнения параллельных участков кода, при котором выполнение ветвей завершается в специально обозначенном месте для того, чтобы в следующей точке продолжить последовательное выполнение. Параллельные участки могут разветвляться рекурсивно до тех пор, пока не будет достигнута заданная степень гранулярности задачи.

Различные реализации FJM обычно управляют *задачами*, *волокнами* или *легковесными нитями*, а не *процессами* уровня операционной системы, и используют *пул потоков* для их выполнения.

На слайде представлены три участка программы, которые потенциально разрешают параллельное исполнение различных блоков. Последовательное выполнение показано сверху, в то время как его Fork-Join эквивалент снизу.

Легковесные нити, используемые в Fork-Join программировании, обычно имеют свой собственный планировщик, который управляет ими, применяя схему пула потоков.

14 слайд

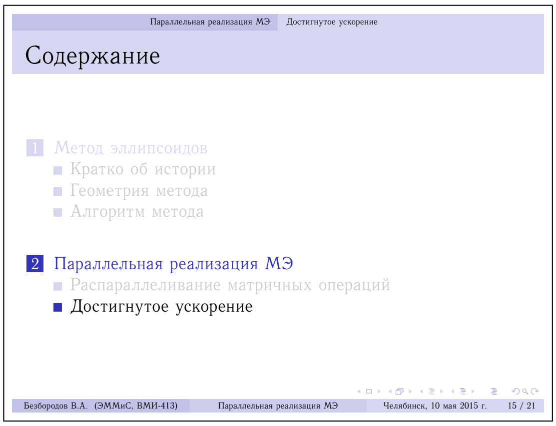
Метод эллипсоидов активно использует матричные операции, вычислительная сложность которых варьируется от $O(n^2)$ (сложение, вычитание, транспонирование) до $O(n^3)$ (умножение). Ускорив выполнение этих операций, можно добиться ускорения выполнение всего метода.

Для многих методов матричных вычислений характерным является повторение одних и тех же вычислительных действий для разных элементов матриц. Данный момент свидетельствует о наличии *параллелизма по данным* при выполнении матричных расчетов и, как результат, распараллеливание матричных операций сводится в большинстве случаев к разделению обрабатываемых матриц между потоками. Выбор способа разделения матриц приводит к определению конкретного метода параллельных вычислений; существование разных схем распределения данных порождает целый ряд *параллельных алгоритмов матричных вычислений*.

Наиболее общие и широко используемые способы разделения матриц состоят в разбиении данных на *полосы* (по вертикали или горизонтали) или на прямоугольные фрагменты (*блоки*).

При *ленточном* разбиении каждому потоку выделяется то или иное подмножество строк (*горизонтальное разбиение*) или столбцов (*вертикальное разбиение*) матрицы. Разделение строк и столбцов на полосы в большинстве случаев происходит на *непрерывной* (*последовательной*) основе.

15 слайд

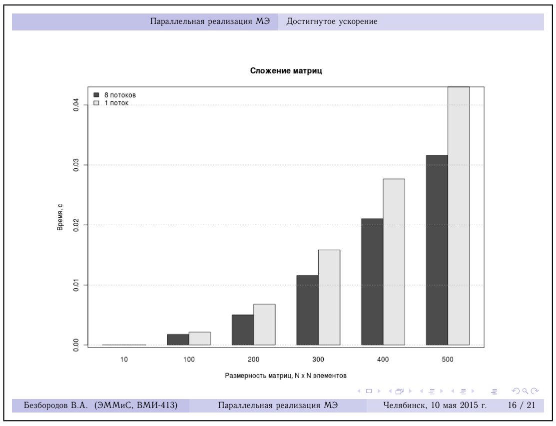


Для разработанного класса необходимо провести анализ эффективности, доказывающий его преимущество перед последовательным выполнением.

Алгоритмы параллельных матричных операций, основанные на ленточном горизонтальном разбиении матрицы, обладают хорошей «локализацией вычислений», т.е. каждый поток параллельной программы использует только «свои» данные, и ему не требуются данные, которые в данный момент обрабатывает другой поток, нет обмена данными между потоками, не возникает необходимости синхронизации. Это означает, что практически не существуют накладные расходы на организацию параллелизма (за исключением расходов на создание/завершение потоков), и можно ожидать линейного ускорения.

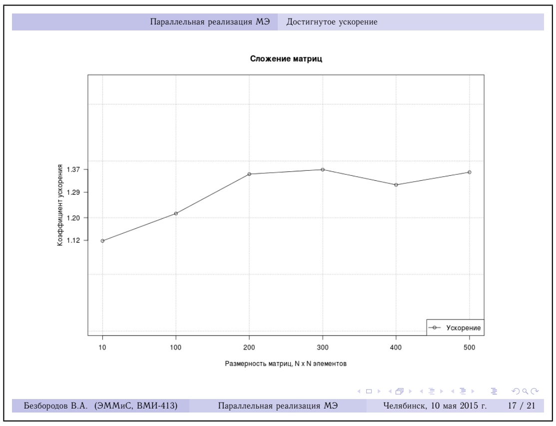
Рассмотрим ускорение, которое удалось получить на практике.

16 слайд



Задача сложения матриц обладает сравнительно невысокой вычислительной сложностью – трудоемкость алгоритма имеет порядок $O(n^2)$. Такой же порядок – $O(n^2)$ – имеет и объем данных, обрабатываемый алгоритмом сложения. Время решения задачи одним потоком складывается из времени, когда процессор непосредственно выполняет вычисления, и времени, которое тратится на чтение необходимых для вычислений данных из оперативной памяти в кэш память. При этом время, необходимое для чтения данных, может быть сопоставимо или даже превосходить время счета.

17 слайд

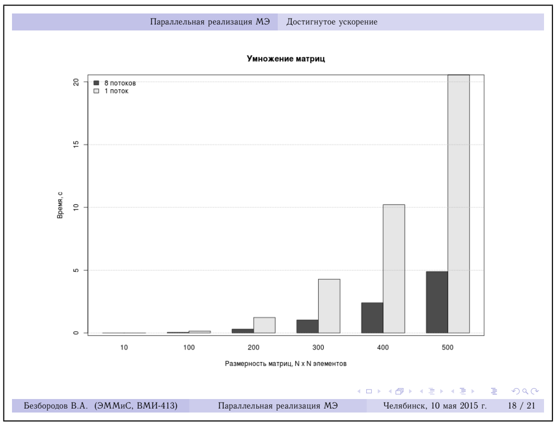


На слайде представлено достигнутое ускорение для операции сложения матриц различных размерностей.

Под *ускорением* выполнения операции понимается отношение времени выполнения операции в многопоточном режиме ко времени выполнения той же операции в однопоточном режиме.

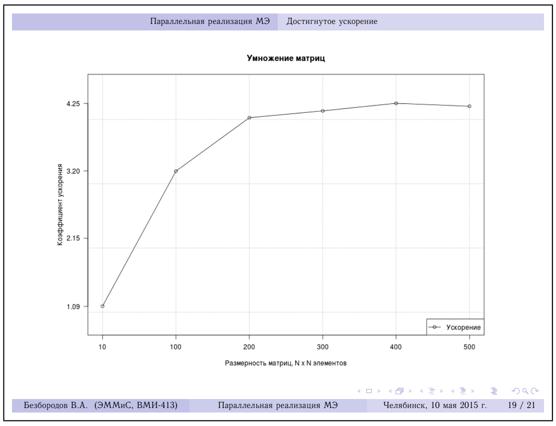
Несмотря на меньшую вычислительную сложность, время работы параллельного алгоритма сложения матриц превосходит время выполнения однопоточной версии в среднем всего в 1.3 раза. Этот эксперимент можно рассматривать, как подтверждение предположения о том, что значительная часть времени тратится на выборку необходимых данных из оперативной памяти в кэш процессора.

18 слайд



На слайде представлено достигнутое ускорение для операции умножения матриц различных размерностей.

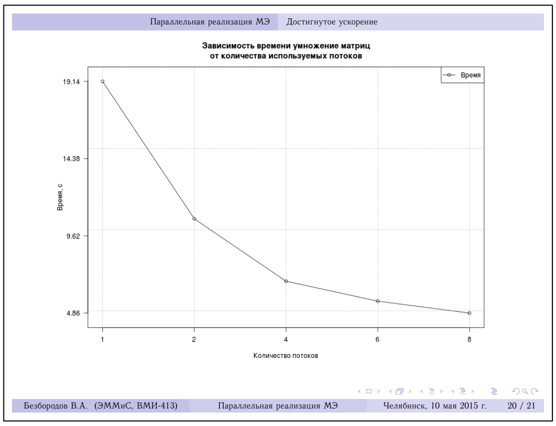
19 слайд



Ускорение имеет существенный порядок из-за того, что сложность операции умножения $O(n^3)$.

Проведем другой эксперимент. Зафиксируем и положим равной 500×500 размерность матриц–операндов. Будем выполнять операцию умножения матриц, применяя каждый раз различное количество потоков, чтобы определить поведение функции времени.

20 слайд



Из рисунка видно, что линейное увеличение количества используемых потоков приводит к нелинейному падению времени выполнения операции умножения. Из анализа результатов эксперимента также следует, что использование числа потоков большего, чем аппаратно поддерживается оборудованием, не приведет к дальнейшему падению функции времени. Такое замедление будет возникать из-за частых переключений планировщика потоков для симуляции параллелизма.

21 слайд